SAND2015-6869 TR

Dakota Software Training

Input Syntax: Configuring Dakota Components

http://dakota.sandia.gov

Exceptional service in the national interest

Module Learning Goals

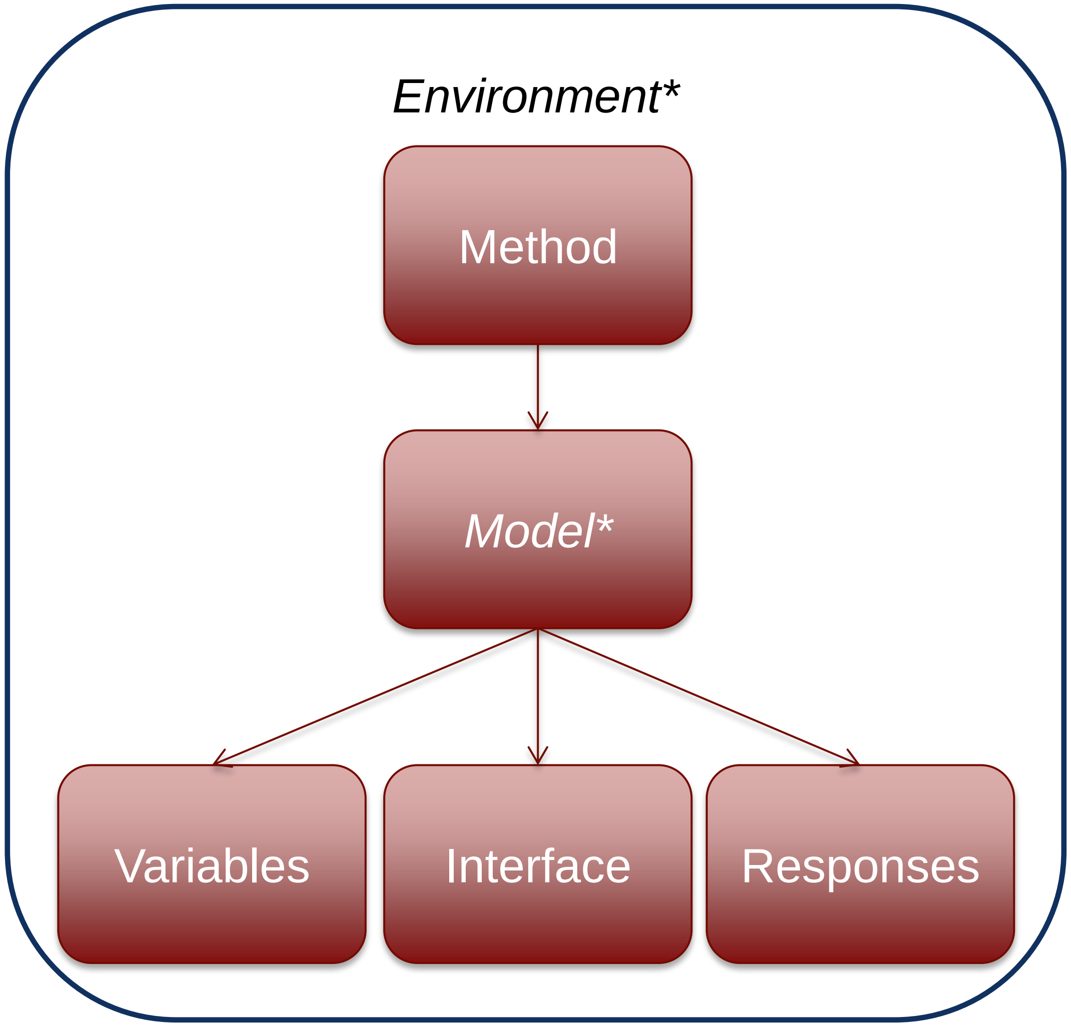
- Develop an accurate “mental model” of Dakota components
- Understand how to configure Dakota components using a Dakota input file
- Become familiar with the Dakota Reference Manual

DAKOTA COMPONENTS

Dakota Blocks

Every Dakota study is composed of **six** types of blocks

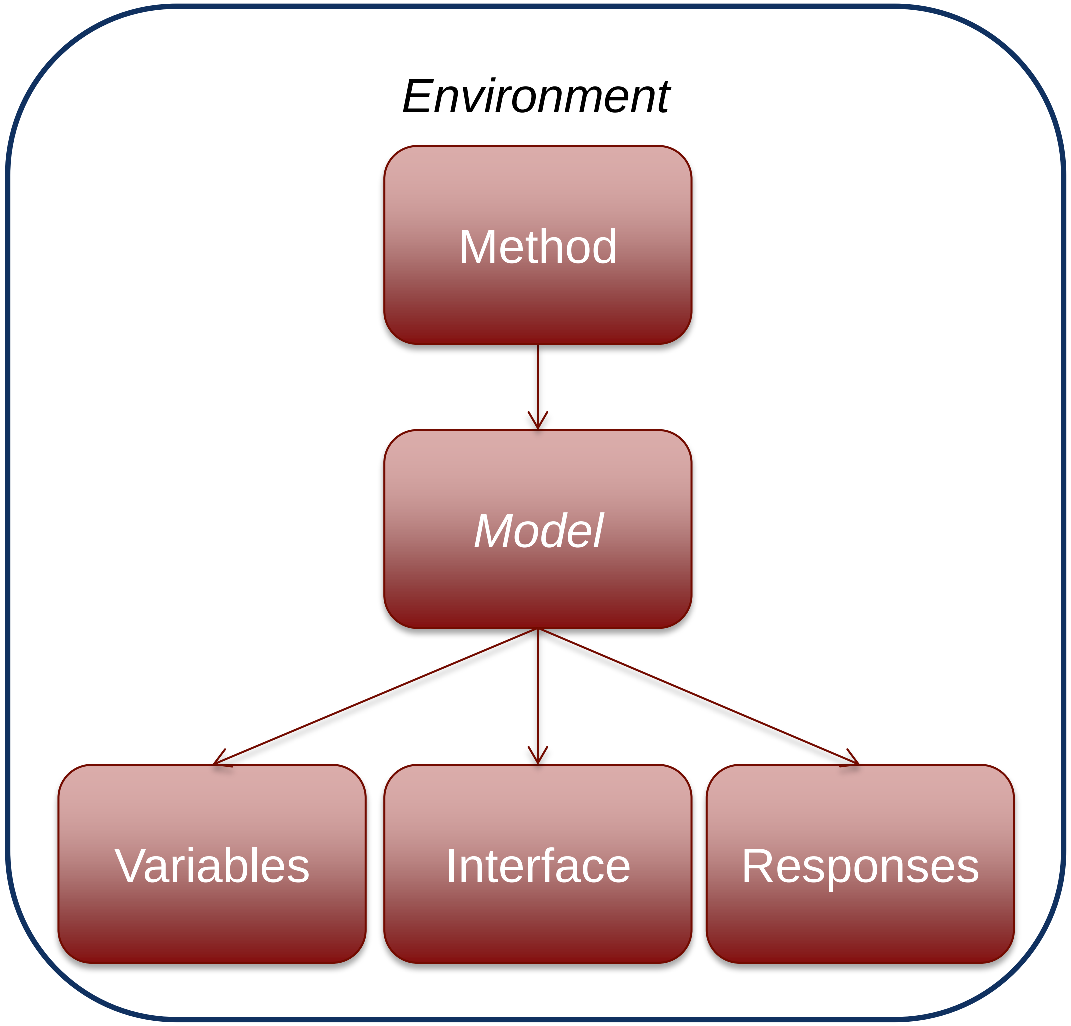
- *Environment**
- Method
- *Model**
- Variables
- Interface
- Responses



**Optional*

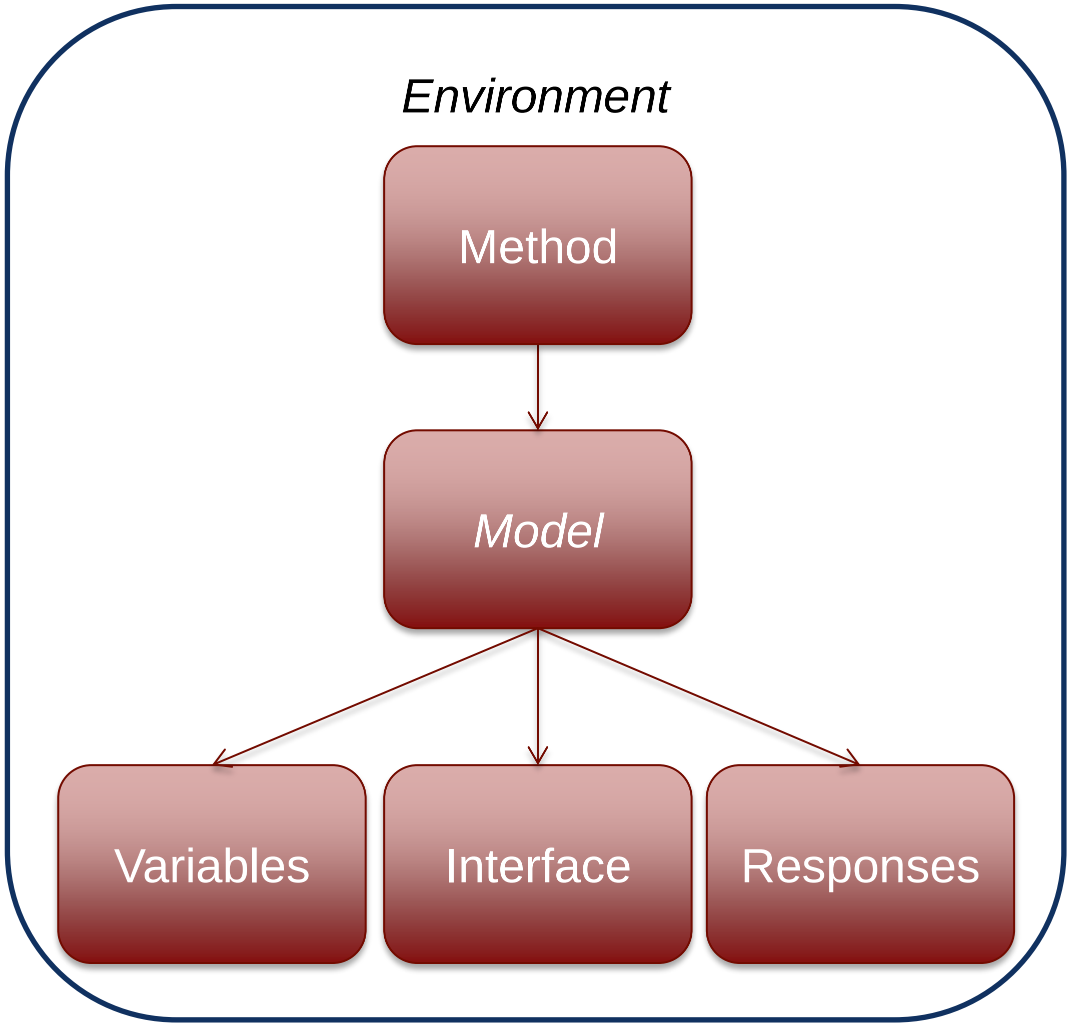
Environment: General Study Settings

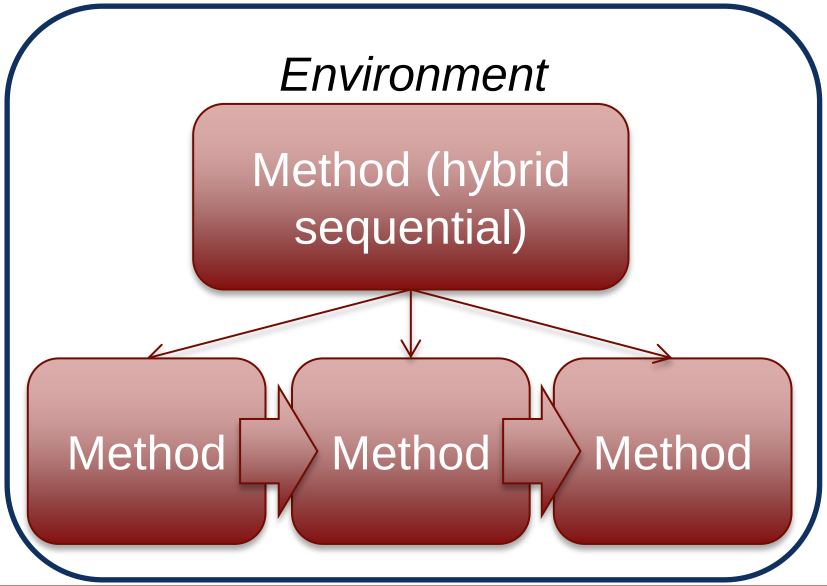
- All other blocks are embedded in one environment block
- Example settings
  - Output precision
  - Name of output & error files
- Selects top-level method, if more than one is present
- *At most* one environment specification is permitted
- Environment specification is *optional*



Method: What Dakota Does

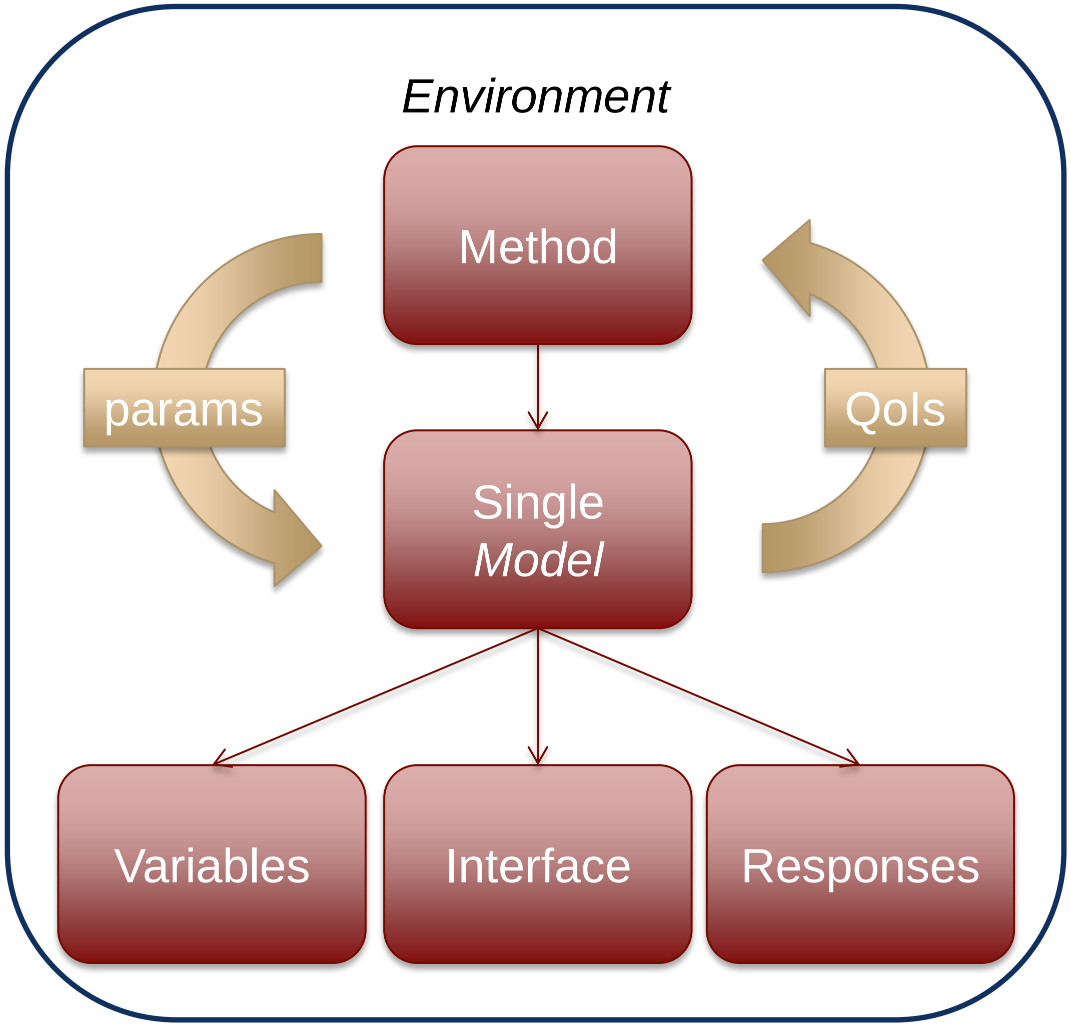
- Specifies an algorithm, e.g., sampling, along with its options
- Invokes the model for a variables-to-responses map
- At least one method block is *required;* multiple may be specified in complex studies

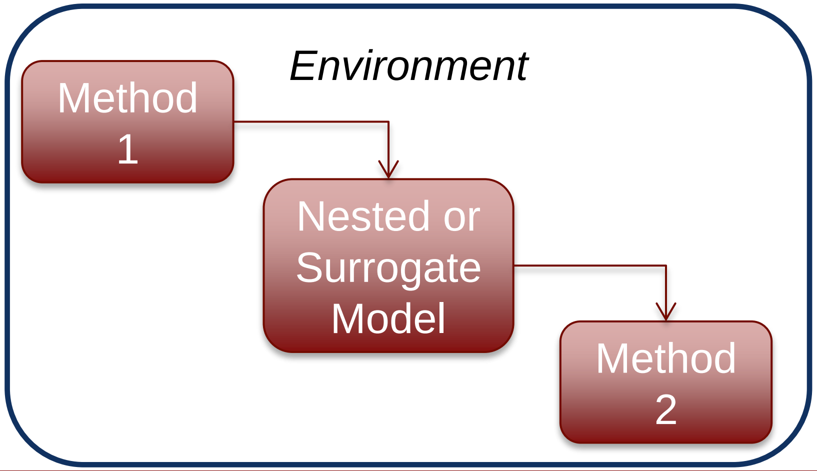




Model: Maps Variables to Responses

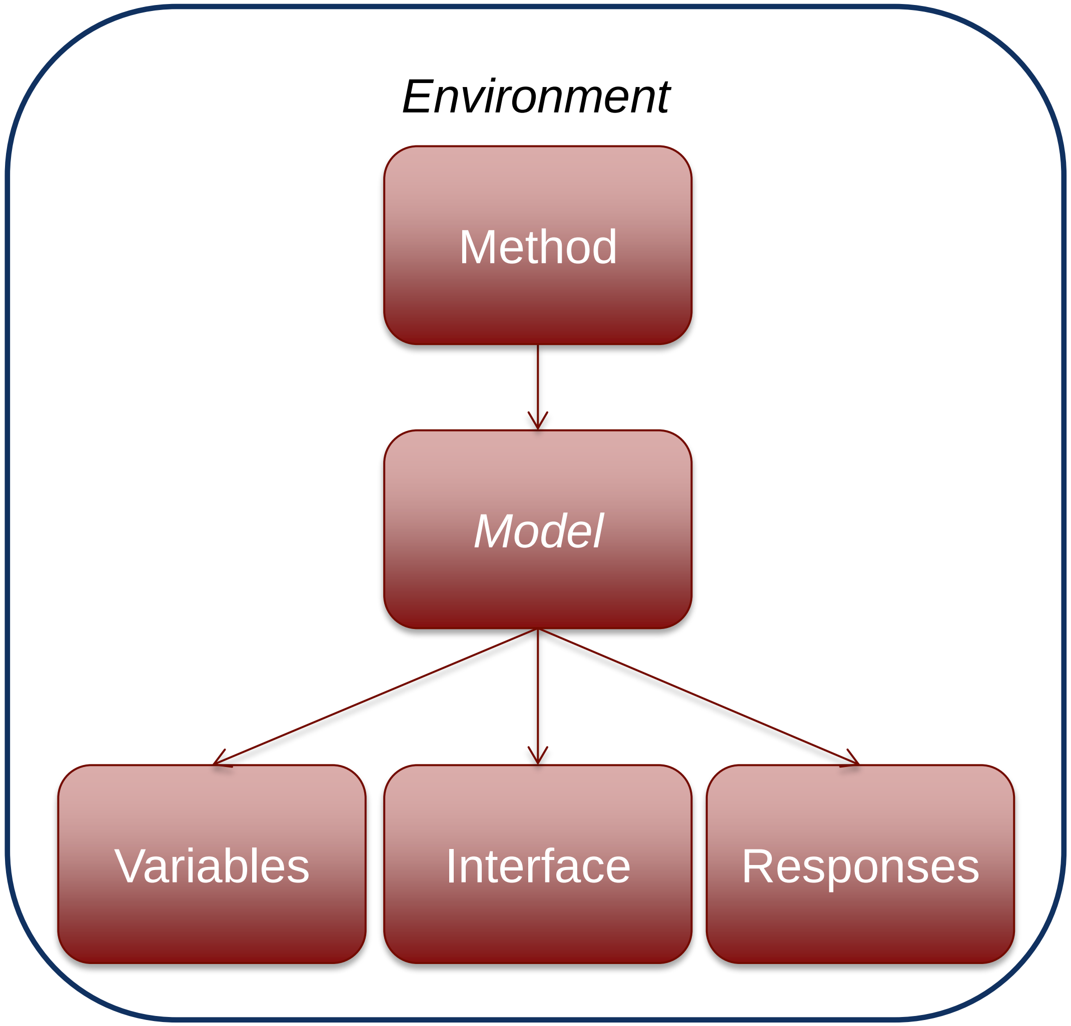
- A model block fulfills requests from a method to compute responses from variables
- Mapping can be through a simulation interface, surrogate model, nested model/method
- Model is *optional* in simple studies: single model is assumed for one vars, interface, resp





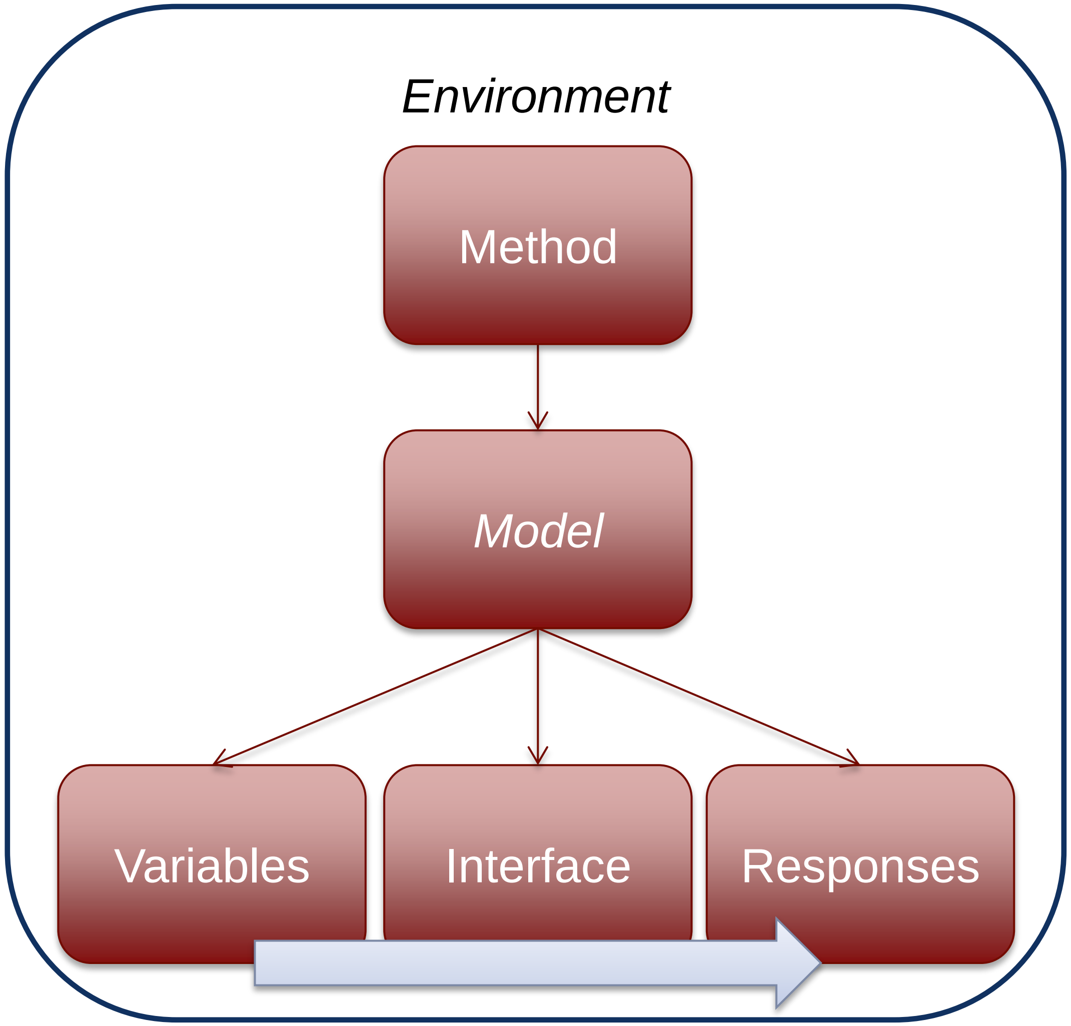
Variables: What Dakota… Varies

- Specify the number, type, other properties of the parameters
- Types are categorized along three dimensions
  - Continuous or discrete
  - Real, integer, or string
  - Design, uncertain, or state
- *Design* variables are used in optimization and calibration
- *Uncertain* variables are used in UQ and sensitivity studies
- *State* variables are typically fixed parameters and not varied by the method
- At least one variables block is *required*



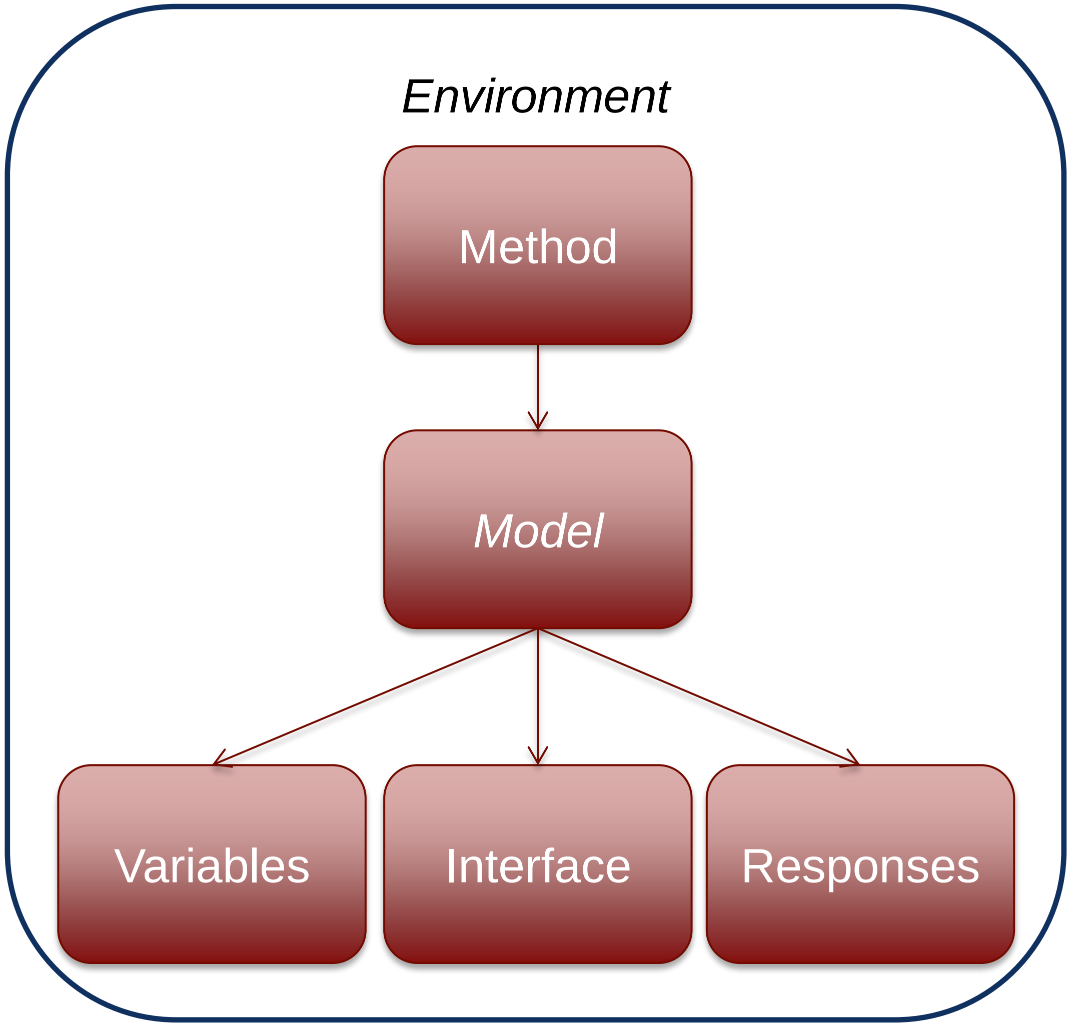
Interface: Communicate with a Simulation

- Specific values of variables are mapped through an interface to obtain responses at those values
- Settings include
  - Path/name to driver
  - Names of files used in I/O
  - Concurrency settings
- At least one interface specification is *required*



Responses: Your Simulation Results

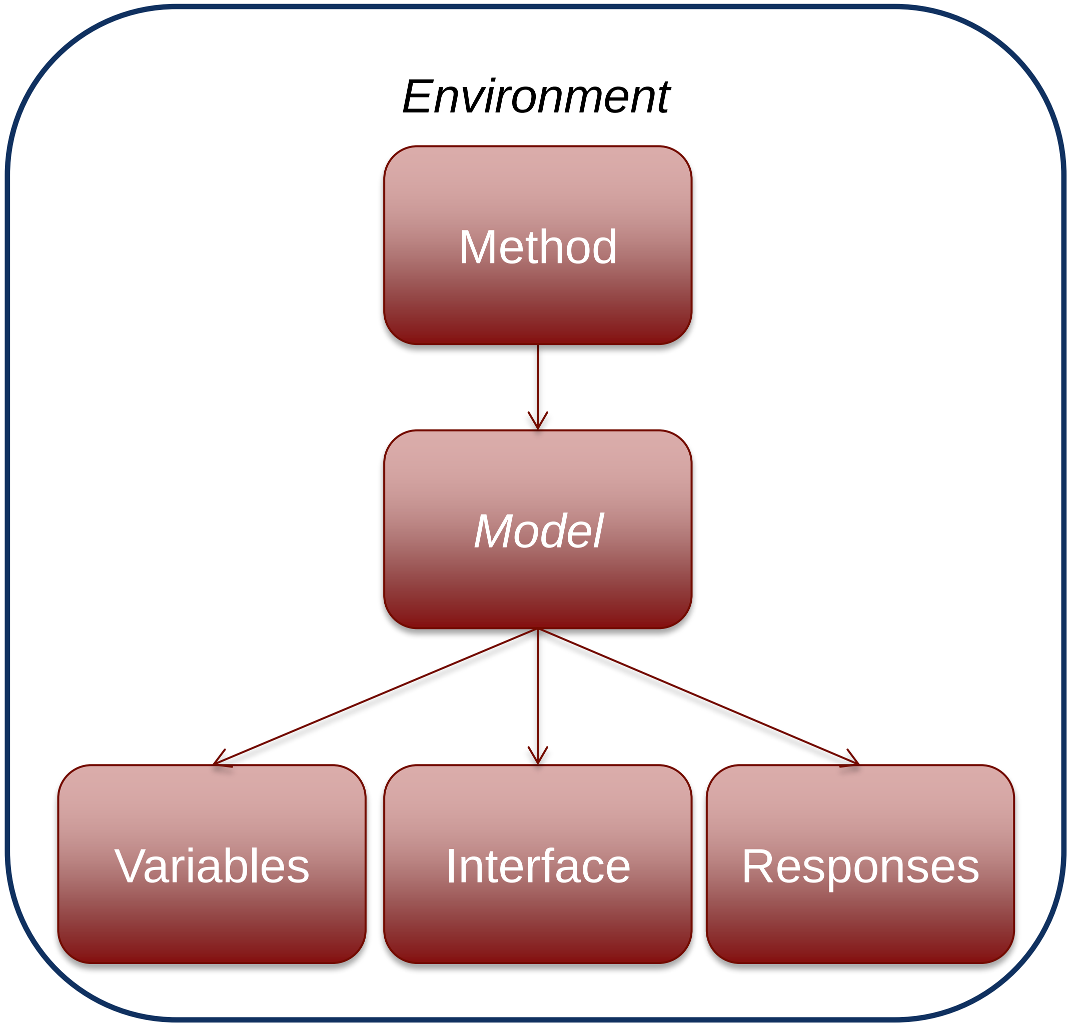
- Data Dakota expects back after setting parameters and running the interface
- Categorized based on usage
  - Objective functions→ optimization
  - Calibration terms → calibration
  - Response functions → SA & UQ
- At least one responses block is *required*



Dakota Blocks In Summary

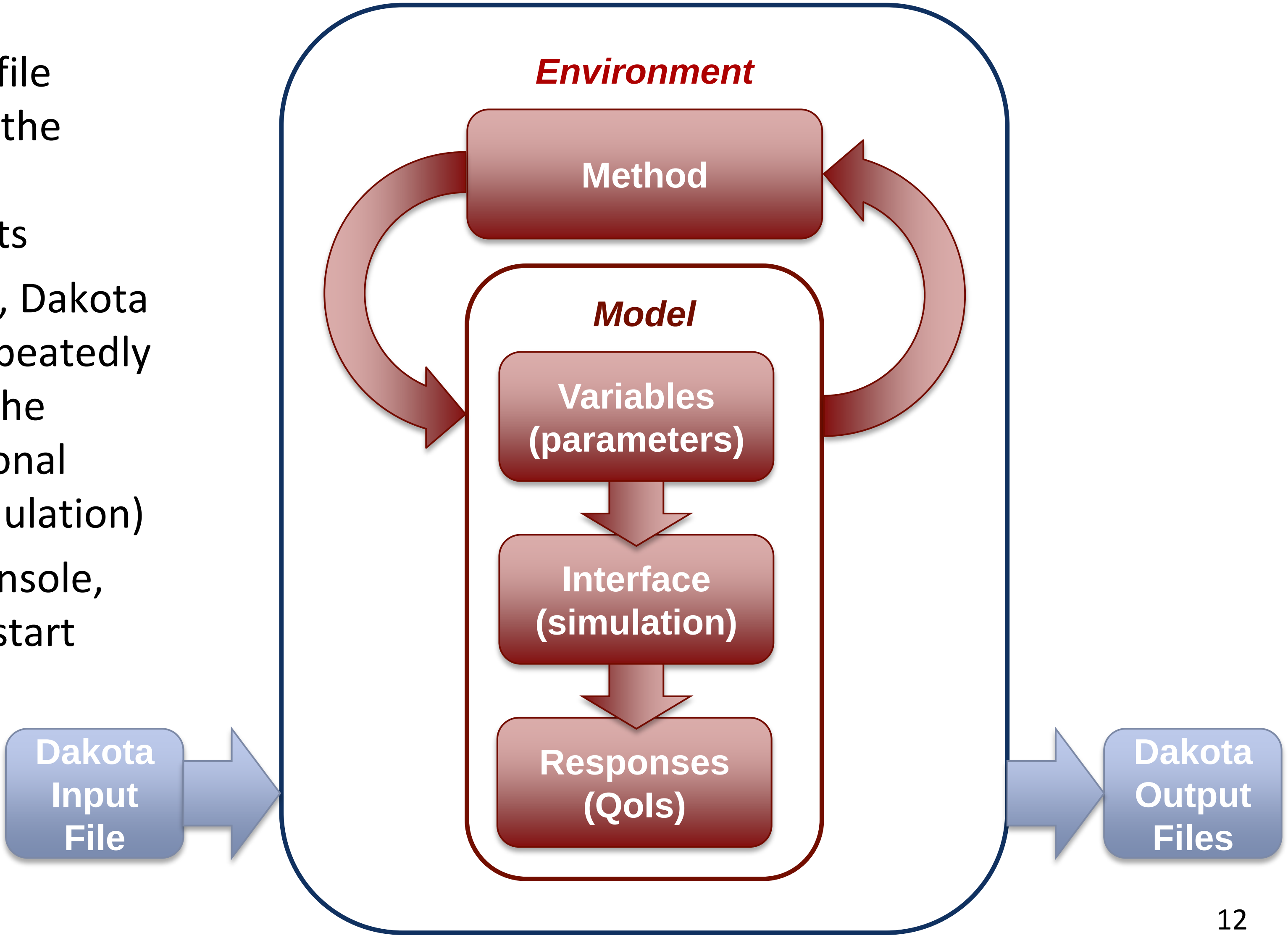
"In each iteration of its algorithm, a **method** block requests a **variables**-to-**responses** mapping from its **model**, which the model fulfills through an **interface***"

*Or surrogate or nested method



Overall Information Flow

- Text input file configures the Dakota components
- At runtime, Dakota method repeatedly evaluates the computational model (simulation)
- Output: console, tabular, restart

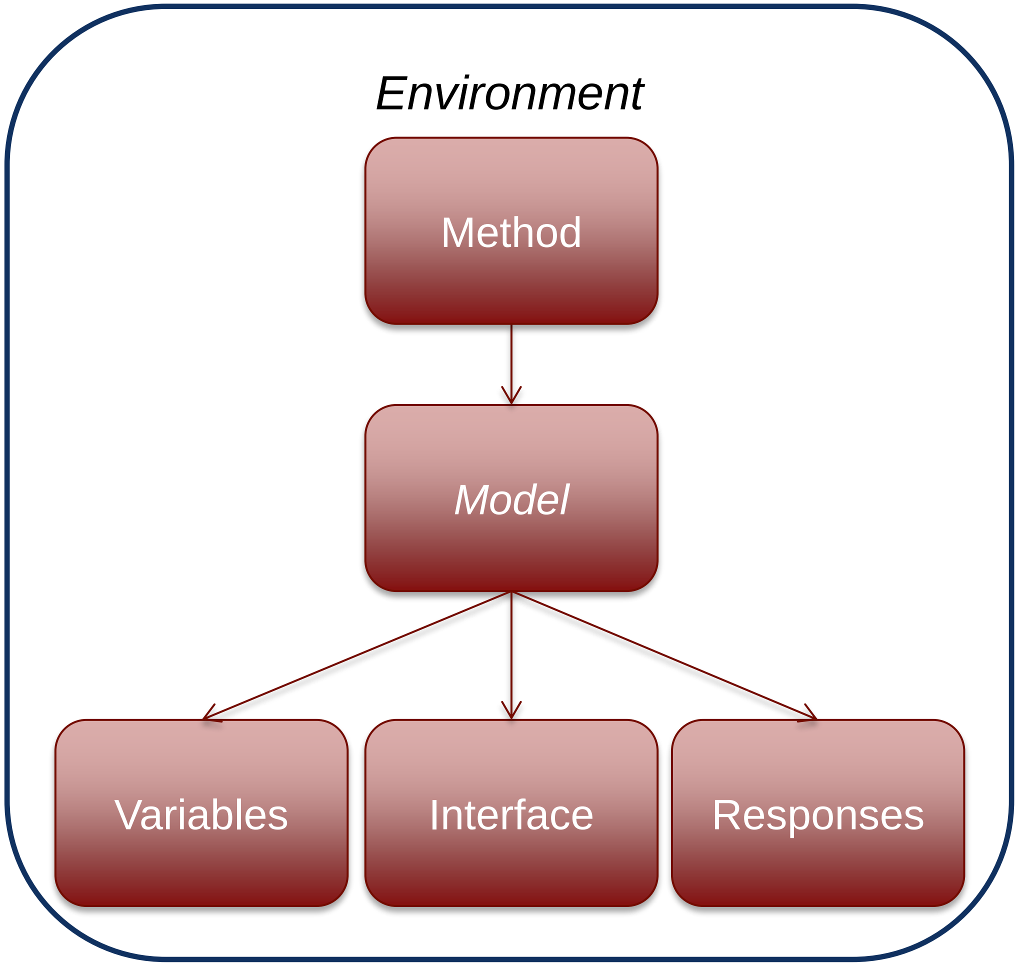


Dakota Input and Components

DAKOTA INPUT FILE

Dakota Input

```
environment
    tabular_data output_precision 1e-16

method
  id_method 'pstudy'
  model_pointer 'pstudy_model'
  centered_parameter_study
    step_vector 0.1 0.1 2.0 10. 1e5 5. 10.
    steps_per_variable 2

model
  id_model 'pstudy_model'
  variables_pointer 'cantilever_vars'
  interface_pointer 'cantilever_inf'
  responses_pointer 'cantilever_resp'
  single

variables
  id_variables 'cantilever_vars'
  active all
  continuous_design = 3
    initial_point  1.0     1.0     20.0
    descriptors    'w'     't'      'L'
  continuous_state = 4
    initial_state  500.0 29.E+6   50. 100.
    descriptors    'p'    'E'     'X'   'Y'

interface
  id_interface 'cantilever_inf'
  fork analysis_drivers = 'driver.sh'

responses
  id_responses 'cantilever_resp'
  response_functions = 3
  descriptors = 'area' 'stress' 'displacement'
  no_gradients no_hessians
```



Input Formatting

```
environment
  tabular_data output_precision 1e-16

method
  centered_parameter_study
    step_vector 0.1 0.1 2.0
                10 1.e5 5. 10.
    steps_per_variable 2

variables
  active all
  continuous_design = 3
    initial_point  2*1.0  20.0
    descriptors    "w"     "t"     "L"
  continuous_state = 4
    initial_state  500. 29.E+6 50. 100.
    descriptors    'p'   'E'   'X'   'Y'

interface
  fork
    analysis_driver = 'driver.sh'

responses
  response_functions = 3
  descriptors = 'area'
                'stress'
                'displacement'
  no_gradients no_hessians
```

Rules

- “Flat” text only
- Whitespace is ignored
- Comments begin with # and continue to the end of the line
- Keyword order largely unimportant as long as major sections are respected and there is no ambiguity
- Equal signs are optional
- Strings surrounded by single or double quotes (beware of “fancy” quotes)
- Scientific notation is fine
- Repeated values in lists: N*Value
- Ranges: Lower:Step:Upper (Range will include the upper bound)
- Unambiguous abbreviations (BUT..)

See the Input Spec section of the Reference Manual

Dakota Input and Components

DAKOTA REFERENCE MANUAL

Dakota Reference Manual

- All permitted Dakota keywords are documented in the Dakota Reference Manual
- The Keywords section of the manual is organized hierarchically and contains
  - Keyword descriptions
  - Usage examples
  - Defaults
  - .. and more
- Let’s take a tour… https://dakota.sandia.gov/content/manuals

Exercise

Goal: Convert a centered parameter study to a different kind of parameter study

1. Copy the example files located in `~/exercises/input/` to a new working directory.
2. Using your favorite text editor, open the copy of `dakota_cantilever_centered.in` that you made. Change it to a different kind of parameter study (**Hint:** *Use the Reference Manual!*)
3. Run Dakota using your modified input file:

```
dakota -i dakota_cantilever_centered.in
```

Does it do what you expect based on the method description in the Reference Manual? Discuss any challenges with your neighbor.